Expertise
and insight
for the future

Tai Pham

Safety System for Material Handling Vehicle prototype

Metropolia University of Applied Sciences

Bachelor of Engineering

Electronics

Bachelor's Thesis

30 September 2019

Author	Tai Pham
Title	Safety System for Material Handling Vehicle prototype
Number of Pages	40 pages + 2 appendices
Date	30 September 2019
Degree	Bachelor of Engineering
Degree Programme	Electronics
Professional Major	
Instructors	Janne Mäntykoski, Senior Lecturer

The goal of this project was to design a safety system for Material Handling Vehicle (MHV) prototype, which defines a safety distance that can change depending on the moving speed. The system will stop a prototype from moving forward when an object is detected within that safety distance.

Based on theoretical knowledge of ultrasonic and photoelectric sensor concept, HC-SR04 sensor for distance measuring and H206 sensor for speed calculating were used to implement the system. Several additional modules were also required to build the complete prototypes. The signals from the sensors are processed by Arduino Uno microcontroller board. Programs were written in C language to control the movement of prototype via L298N motor driver and to determine safety distance for the system. Information of distance and speed will be shown on Liquid-crystal Display (LCD) 1602.

Two prototypes were created to demonstrate two types of MHV, which are human driven one and Automated Guided Vehicle (AGV). The safety system can successfully separate different safety distances for different speed modes for both prototypes. Therefore, with further developments, this project can be applied to increase the safety of MHV, as well as minimize the accident rate in warehouse industry.

Keywords	Safety distance, ultrasonic sensor, speed sensor, IR sensor, joystick, line follower, PWM

Contents

Appendices

Appendix 1. Joystick-Control Prototype code

Appendix 2. Line Following Prototype code

List of Abbreviations

AGV	Automated Guided Vehicle
IR	Infrared Radiation, Infrared Light
ISR	Interrupt Service Routine
LCD	Liquid-crystal Display
LED	Light-emitting Diode
LiDAR	Light Detection and Ranging
Li-Po	Lithium Polymer
MHV	Material Handling Vehicles
PID	Proportional-Integral-Derivative
PWM	Pulse Width Modulation

1 Introduction

The workplace safety management is considered as one of the most important factors to all industries, as it benefits the welfare of both employees and employers. According to Statistics Finland, in 2010, storage was mentioned as one of the riskiest industries as the risk of death was 4.6 over 100,000 employees [1]. Since global trading continuously increases, warehousing services have played a vital role in the storage and transportation of goods. Besides, the arrival of MHV has supported and improved several features in warehousing services, such as shorten delivery time and reducing handling costs in the supply chain. Generally, there are two main types of MHV, which are the human-driven vehicle and AGV. The former's movement is operated by human, while the latter is programmed to move without human driver. The routes of AGV are non-wire guidance mechanisms, such as lasers and magnetic tape. Depending on the design of the warehouse as well as the type of the load, these vehicles travel with different velocity. However, these vehicles are still a major cause of accidents at the warehouse. Therefore, along with conducting proper training for vehicles operators and pedestrians, new technologies have been researched and applied to increase the safety of MHV and minimize the accident, as well as injury rate at the warehouse.

The main goal of this study was to design a safety system for two prototypes. The first one is a robot with directional control, which represents a human-driven vehicle. The second one is a line follower robot, which represents an AGV. Particularly, the moving robot stops when an object is detected within a safe distance, which is dependent on the speed of the vehicle. The faster the vehicle travels, the longer the safety distance is.

The second chapter covers the theoretical knowledge on ultrasonic energy and photoelectric effect, which are applied in this study for distance and speed measurement. It also discusses the basic concept of line follower robot and Proportional-Integral-Derivative (PID) control. Chapter 3 offers key information about the main components which are used to build two prototypes. Chapter 4 focuses on the development and implementation of the design of two prototypes, the details of components wiring. Finally, chapter 5 presents the conclusion for this study, evaluates performance of the prototypes and suggests possible improvement for later version.

2 Theoretical Background

This chapter delivers a short description of the physical principles of the sensors which are used in this study. It is useful for a better understanding of the characteristics, functions, advantages and disadvantages of these sensors. Selection of specific sensors and their performance are discussed in following chapters.

2.1 Concepts of Ultrasonic Sensors

For many industrial applications such as process control, robotics, security system and especially transportation traffic control, the distance measurement of an object's position to a selected reference is essential. In this study, by measuring the distance accurately, it is possible to determine the vehicle's safety distance to pedestrians or potential hazards in the warehouse, which minimizes the possibility of accidents and injuries to happen.

For non-contact distance, Fraden [2,314] claims that a measurement design can be made with an active sensor (an external source of power is the requirement to operate) which transmits a pilot signal and receives a reflected signal. The transmitted energy in this study is ultrasonic, which is an acoustic (sound) energy in the form of waves transmitted at a frequency beyond the hearing range of human (20 kHz per second). Ultrasonic energy's transmission and reception are fundamental for many automation applications such as distance measurement, position changes and velocity detectors. [2,314.]

When ultrasonic waves encounter an object, part of their energy is reflected in a diffuse manner, which may approach 180° regardless of the coming direction. Based on the Doppler effect, in case of the object moves, the reflected wavelength frequency is different from the transmitted waves one. [2,314.]

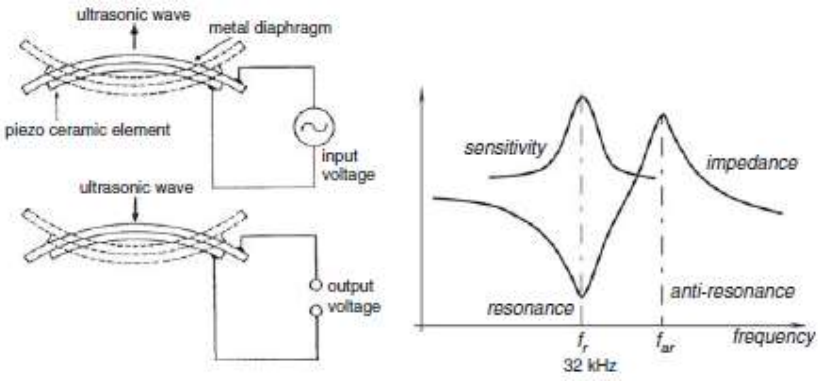

Figure 1. Piezoelectric ultrasonic transducer and its impedance characteristic [2,315,316]

Working principle of the ultrasonic sensor is showed in figure 1 and figure 2. When an input voltage is applied to the sensor transducer, it flexes the ceramic element then a burst of the ultrasonic sound waves is generated and transmitted at the frequency around 32kHz (Figure 1). When the waves encounter an obstacle, they reflect and flex the ceramic, where produce an echo voltage signal (Figure 1).

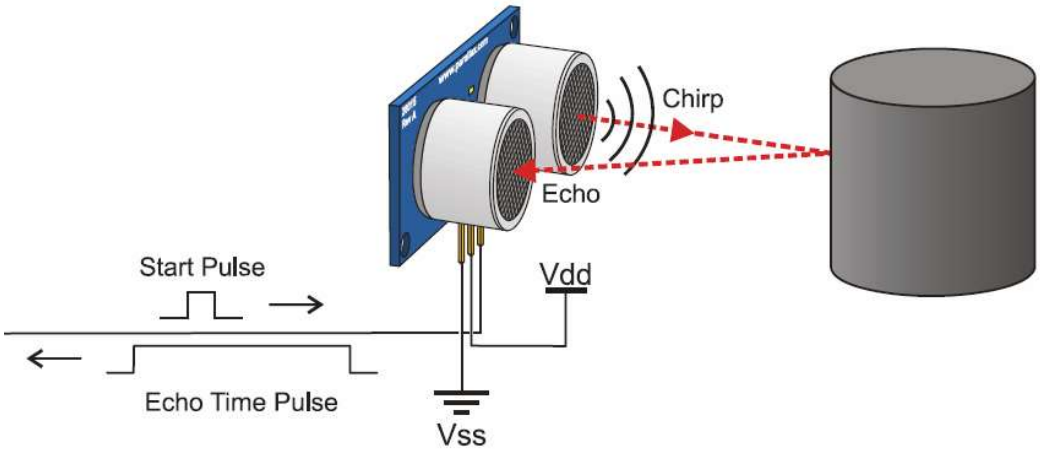

Figure 2. Ultrasonic sensor principle operation [3]

The distance from the object is proportional to the timespan between emitting the signal and receiving the echo. As can be seen in figure 1, to improve the efficiency, driving oscillator frequency should be modified to the resonant frequency f_r, where the sensitivity and efficiency of the piezoelectric ceramic element are the best.

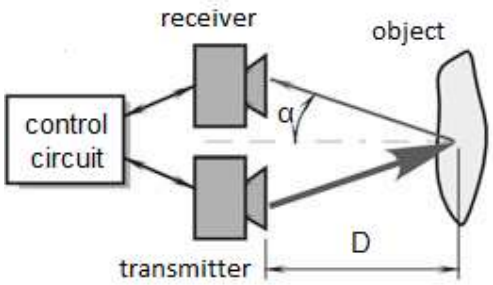

Figure 3. The basic arrangement of ultrasonic distance measurement. [2,315]

The distance D can be calculated based on this formula:

$$D = \frac{1}{2} v t \cos(\alpha) \quad (1)$$

Where v is the speed of sound (because ultrasonic waves propagate in the media with the speed of sound; v = 343,2 m/s at 20°C), t is the timespan that ultrasonic waves travel from the transmitter to the object and back to the receiver, and α is the angle which can be seen in figure 3. If the distance D to object is far longer compared to the gap between a transmitter and a receiver, then cos(α) ≈ 1 [2,315].

Propagation with the speed of sound is an advantage of ultrasonic waves compared to the microwaves which travel with the speed of light. Speed of sound is much slower, so time t is much longer. Thus, it is easier and cheaper for measuring with ultrasonic waves. Additionally, ultrasonic waves are also unaffected by the color of the material they are sensing and completely insensitive to hindering factors such as light, dust, smoke, mist, vapor, etc [2,315].

However, the speed of sound varies depending on temperature and humidity. Therefore, to improve the accuracy, ambient temperature and humidity should be measured for the distance calculation.

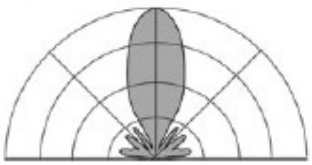

Figure 4. Ultrasonic sensor directional diagram [2,316]

The figure above shows another limitation when using the ultrasonic sensor. It illustrates the sensor's effective angle of operation. The sensor will provide the most accurate reading signal when a detected object is directly in front of it within 30 degrees window.

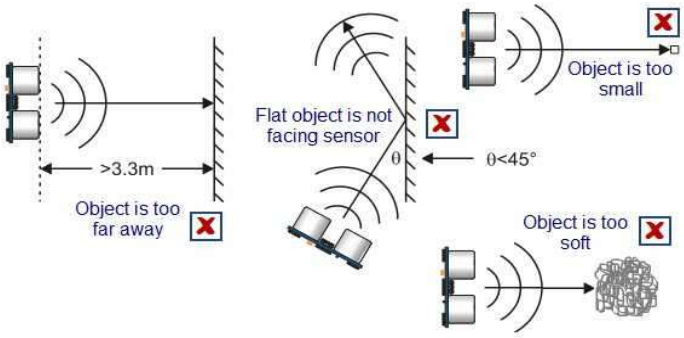

Figure 5. Ultrasonic sensor limitations [4]

Additionally, other limitations of the ultrasonic sensor are shown in the figure above. Each type of ultrasonic sensor offers a different effective range for operation, if the object is out of that range, the reading signal is not precise. In case the object is not perfectly in front of the sensor, the echo signal may be deflected and does not return to the receiver. Moreover, if the object is too small or too soft, the ultrasonic waves may not be reflected.

2.2 Concepts of Photoelectric Sensors

For many industrial applications in packaging, textile, paper industries and especially automated processes, photoelectric sensors have become crucial elements. Increasing

demand for safety, process efficiency and precision in sensing is the reason that photoelectric sensors are widely used. In 2015, Europe was the region with highest photoelectric sensor market revenue share and is expected to hold highest regional market share in future years, thanks to the rising demand of industrial automation in this region [5].

Photoelectric sensors detect objects without involving physical contact using light, electronic signal evaluation and amplification, and producing output state. A photoelectric sensor is constructed of an emitter (light source) for emitting light and a receiver for receiving light. When emitted light is incident or reflected by the sensing objects, the amount of light changes and reaches the receiver. The receiver senses this change and transfers it to an electrical output.

2.2.1 Light Source

Pulse modulated light that emits light repetitively at fixed intervals is commonly used by photoelectric sensors as a light source (Figure 6). It increases the sensing range while decreasing the interference of ambient light.

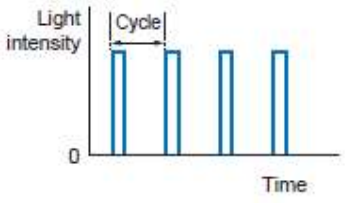

Figure 6. Pulse modulated light intensity [6]

Similar to the working principle of a radio receiver and radio station, photoelectric sensors can distinguish the modulated light and ignore the ambient light. They commonly use modulated light that ranges in the light spectrum from visible green to visible infrared as a light source. The wavelength of those lights can be seen in figure 7 and the detail range of interest light is shown in table 1. It shows that infrared light with the wavelength λ between 850 nm and 1050 nm has stronger intensity than other light.

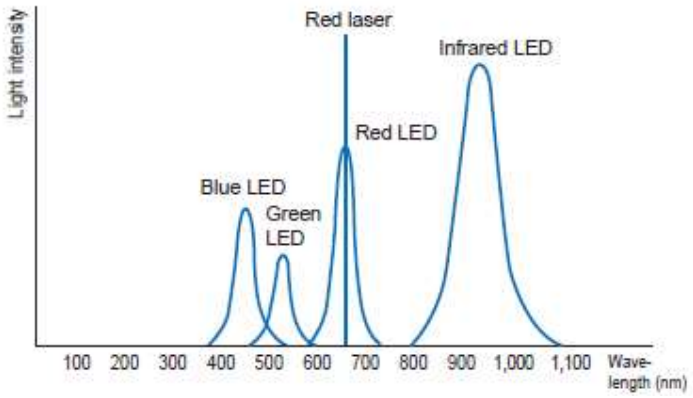

Figure 7. Electromagnetic radiation spectrum [6]

Photoelectric sensors most often use infrared light with the wavelength λ = 880 nm as a light source. Besides, red light with λ = 660 nm and infrared light with λ = 950 nm are also used in some specific cases [7,13]. According to a training manual offered by ifm electronic, there are several advantages of using infrared light as emitted light. When the same current is applied, the transmitter diodes for infrared light release more radiation, which means higher efficiency is acquired. Moreover, due to its wavelength is longer than the diameter of dust particles, transmitting of infrared light has no trouble with dust and soil. [7.]

Wavelength range	Radiation designation
400 nm - 440 nm	light - violet
440 nm - 495 nm	light - blue
495 nm - 558 nm	light - green
558 nm - 640 nm	light - yellow
640 nm - 800 nm	light - red
800 nm - 1400 nm	Infrared light (IR) - A
1.4 µm - 3,0 µm	IR - B
3.0 µm - 1000 µm	IR -C

Table 1. The wavelength range of interest light according to DIN 5031 [7,13]

There are also other types of sensors, for example, Mark Sensors, which use non-modulated light as a light source. In contrast to pulse-modulated light, non-modulated light emits an uninterrupted beam of light at a specific intensity (Figure 8).

Figure 8. Non-modulated light intensity [6]

The advantage of non-modulated light is offering fast response time for the sensors while the limitations are short sensing range and susceptibility to ambient light interference [6].

2.2.2 Technology

Based on technology, photoelectric sensors are classified into through-beam, diffuse-reflective and retro-reflective. Depending on the target, the best technique is chosen. Some objects are highly reflective while others are hazy. In some situations, color-changing detection and scanning distance are also a consideration when selecting the technique. Some sensing methods operate well when the object is closer to the sensor, and others perform better at a greater distance. [8, 84.]

i Through-Beam Sensors

A through-beam sensor consists of separate emitter and receiver which are installed opposite each other. This allows the greatest possible amount of pulse-modulated light from the emitter to arrive at the receiver. When an object interrupts the direct path of the light beam between emitter and receiver, it triggers the receiver's output to change state. The receiver's output returns to the normal state when the light path is clear. Through-beam sensing technique is demonstrated in figure 9.

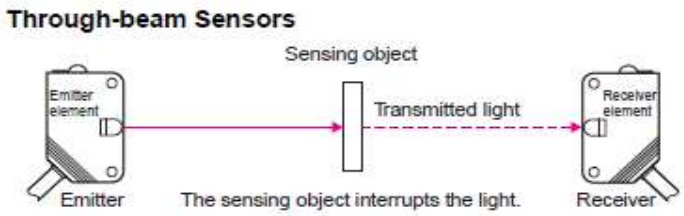

Figure 9. Through-beam sensing method [6]

Besides, slot sensors are a through-beam version that constructed with an integrated emitter and receiver (Figure 10). This is the type that is used in this study, combined with a round grid plate to calculate the speed.

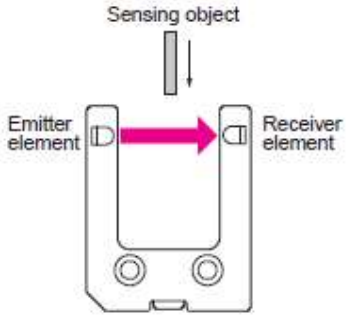

Figure 10. Slot Sensors [6]

Effective beam of photoelectric sensors is the area of the beam's diameter that an object can be sensed. For a through-beam sensor, the diameter of the emitter and receiver lens is the effective beam. It is illustrated in figure 11. The minimum size of the object should be equal to the lens' diameter for better accuracy. [8, 84.]

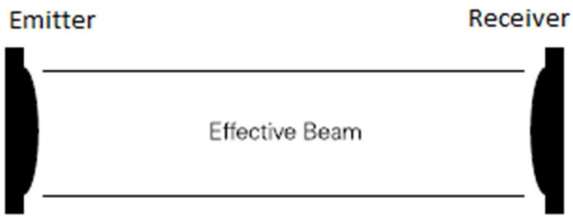

Figure 11. The effective beam of the through-beam sensor [8, 84]

One of the characteristics of this method is long detection distances varying between several centimeters to several tens of meters with stable operation. The other is changes in object path does not affect detecting position. Moreover, object gloss, color, or inclination do not have a huge effect on sensing operation. [6.]

ii Diffuse-Reflective Sensors

A diffuse-Reflective Sensor consists of an emitter and a receiver which are installed into one single unit. The original state is when an object is not present, the emitter strikes the light and there is no reflection of transmitted light to the receiver. The output's state changes when emitted light senses an object and it is reflected to the receiver. Demonstration of this sensing method is shown in the figure below.

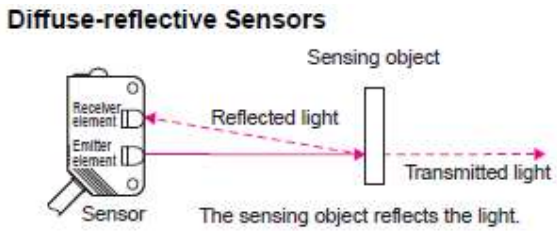

Figure 12. Diffuse-Reflective sensing method [6]

For a diffuse-reflective sensor, the size of the object when detected in the path of light is the effective beam, which is demonstrated in figure 13.

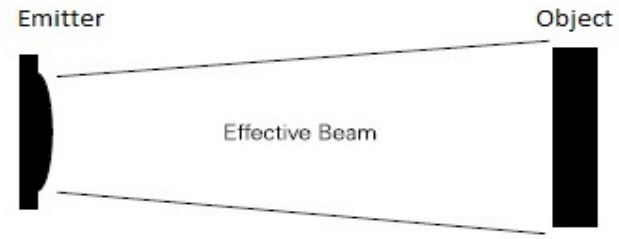

Figure 13. The effective beam of the diffuse-reflective sensor [8, 88]

In practice, the range for accurate detection depends on the characteristics of the object. Object's size and color are two of the factors that have the influence. [7,56.]

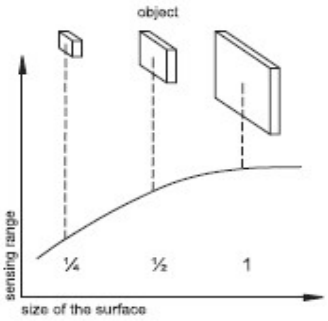

Figure 14. Range and size [7,57]

The fact is a small object reflects less light than a big one so that the range is varied corresponding to the size of the object, which is illustrated in Figure 14.

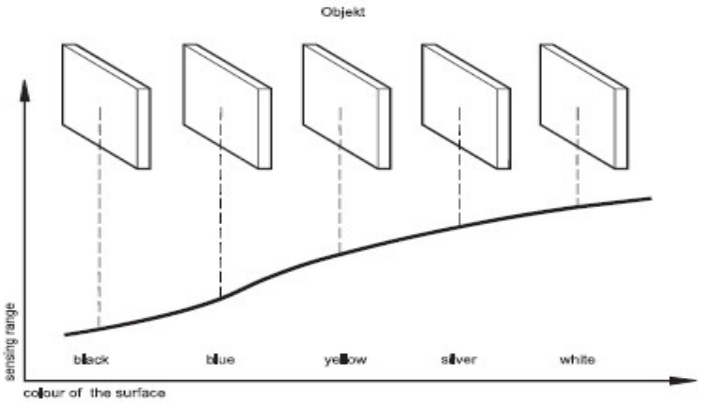

Figure 15. Range and color [7,57]

Generally, the reflective quality of light color is better than dark color. Figure 15 displays the sensing range changes according to the visible light. However, this feature with infrared light is significantly different from that range of visible light [7,57]. Therefore, the sensing range cannot be determined only by the object's color impression.

iii Retro-Reflective Sensors

A Retro-Reflective Sensor consists of an emitter and a receiver which are installed into one single unit like a diffuse-reflective sensor. The original state is when an object is not present, the emitter strikes the straight light to a reflector installed on the opposite side, it is reflected and returns to the receiver. The output's state changes when emitted light is interrupted by an object, its amount reduces when returning to the receiver. Illustration of this sensing technique is shown in the figure 16.

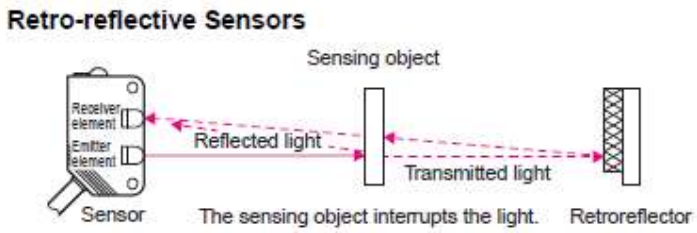

Figure 16. Retro-Reflective sensing method [6]

For a Retro-Reflective Sensor, a tapered area from the sensor's lens to the edges of the reflector is the effective beam (Figure 17). The minimum size of the object should be equal the size of the reflector for better accuracy.

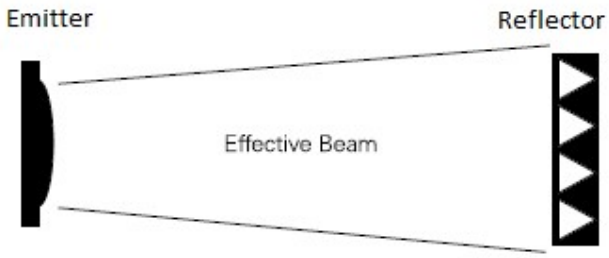

Figure 17. The effective beam of the retro-reflective sensor [8,85]

A reflector can be designed with different size, different shape (rectangular or round) or reflective tape. With a flat reflector, only the emitted light which strikes the mirror vertically is reflected to the receiver (Figure 18).

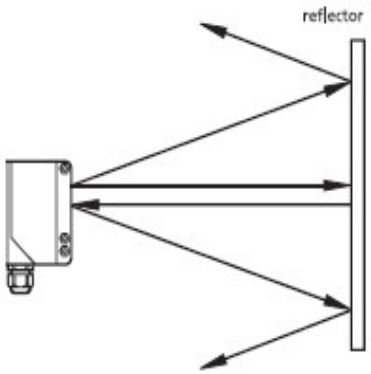

Figure 18. Flat reflector [7,44]

The solution for the receiver to obtain as much light as possible is to use a prismatic reflector (Figure 19). A prismatic reflector contains several tiny prisms, which helps the light beam reflects exactly parallel towards the direction where it arrives from. Considerable margin for the angle that light beam hits the triple prism is ±15°, which is not too inclined. [7, 44.]

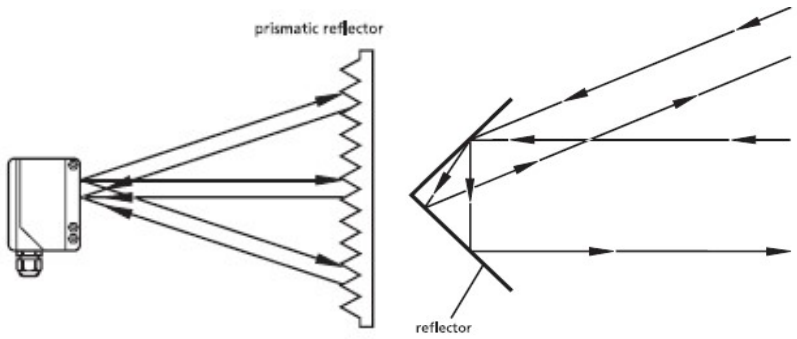

Figure 19. Prismatic reflector [7,44]

As can be seen in figure 19, not only light beam strikes the reflector at 45° returns exactly parallel, but also the sharp-angled beam reflects in parallel.

Generally, the detection distance range for retro-reflective sensors is from several centimeters to several meters. The angle and color of the sensing object do not have a significant effect on the operation. Transparent objects can be detected with this method because the light beam can pass through them twice.

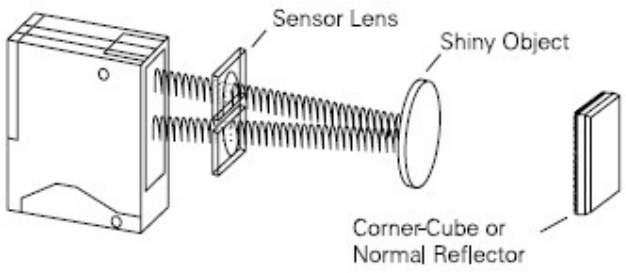

Figure 20. Shiny object confusion with retro-reflective sensor [8, 86]

However, this technique may not able to sense shiny objects because they reflect the light to the receiver. The sensor is incapable of distinguishing the light returns from a reflector or a shiny object (Figure 20). Additionally, another disadvantage of retro-reflective sensors is that they have a dead zone at close distances.

2.3 Concepts of Line Follower Robot

A line follower robot is an electronic device designed to track out and follow a line or a path which is predefined by the user. There are various types of line or track such as physical black or white line on the floor, magnetic markers, embedded lines and laser guide. They vary from simple low-cost technique to the complex expensive system. Depending on the project scope, the suitable option is selected, and appropriate technology and sensors are employed for the robot.

2.3.1 Working Principle

In this study, the prototype is a black line robot follower on a white surface. The robot senses the black line and stays on track while continuously adjusting to correct the wrong moves using the feedback mechanism. This works based on the behavior of light at the white and black surface.

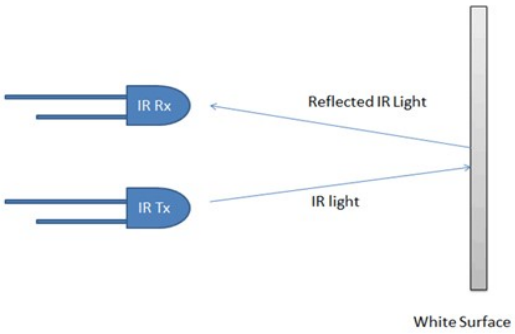

Figure 21. Light is reflected by white surface [9]

In general, the reflective quality of light color is better than dark color. Light is reflected almost completely when it strikes at a white surface (Figure 21) and absorbed totally at a black surface (Figure 22).

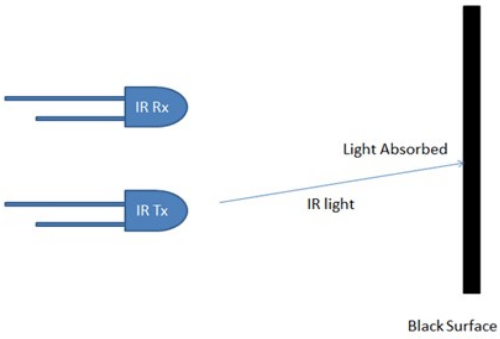

Figure 22. Light is absorbed by black surface [9]

The line follower prototype consists of three units: sensor unit, control unit and driver unit. Sensor unit contains infrared transmitter, infrared receiver, potentiometer and a comparator. The infrared transmitter emits light ray to the surface. When a light ray hits the white surface, it is reflected with high intensity. In case of hitting a black surface, a light ray is absorbed and not reflected. The infrared receiver is a photodiode, which generates analogue voltage corresponding to the intensity of reflected light. This voltage is connected to one terminal of a comparator. Another terminal is connected to a reference voltage, which is set by a potentiometer. Both voltages are compared by a comparator and a digital output signal is created. [9.]

Control unit for this prototype is a microcontroller, which reads the digital signal from comparator output, processes the signal and sends commands to driver unit to control the rotation of the wheels and keep the robot on the line. There are several methods to

process the signals from the sensor unit, and PID control is adopted as a feedback control in this study to increase the performance of the robot.

Driver unit contains a motor driver and two DC motors. Motor driver is used because most often, microcontroller does not provide enough voltage and current to drive the motors.

2.3.2 PID control

PID control is the most common control algorithm used to control feedback loops in many industrial process applications. PID control contains three fundamental terms: proportional, integral and derivative which are computed and summed up to provide optimal result for the process. It can be understood as a control method that tracks the present, the past and the future of the error between the process variable and the setpoint to deliver correction until no error remains. Proportional control depends on the present error, integral control on the accumulation of the past error and derivative control on the forecasting of future error. [10.]

i Proportional control

The proportional part only depends on the current difference between the process variable and the setpoint, which is referred as the current error [11,3]. The output response is determined by multiplying the error by a constant proportional gain Kp. The proportional term is specified by:

$$P = Kp.error(t) \quad (2)$$

For example, if the error term has a magnitude of 10, a proportional gain of 7, the output proportional response of 70 is produced.

Generally, increasing the proportional gain Kp will rise up the speed of the output response. However, if the proportional gain is too high, the process variable starts to oscillate, and the system can be unstable. On the other hand, decreasing the proportional gain results in lowering the speed of the output response. If the proportional gain is too

low, the output response may be too small for the error, and consequently, the steady-state error is reduced but is not eliminated. [10.]

ii Integral control

The integral part depends on both the magnitude and the duration of the error. It is the sum of the instantaneous error over time and provides the output response over time until the error becomes zero. The control signal will be increased when a positive error is detected and decreased with a negative error. This drives the system to eliminate the final difference between the process variable and setpoint, which also called Steady-State error. The integral term is specified by:

$$I = Ki \int_0^t error(t)dt \tag{3}$$

where Ki is the integral gain and it shows that the integral control is based on the error value in the past [11,5].

However, no matter how small the error is, it will change the control signal. Therefore, it may deteriorate the transient response. Moreover, integral windup is a phenomenon that happens when integral action saturates a controller without the controller driving the error signal toward zero. [10.]

iii Derivative control

The derivative part depends on the slope or the error over time and it is related to the future predicted error signal. The derivative action is proportional to the rate of change of the process variable. The derivative term is specified by: [11,6]

$$\mathrm{D} = Kd \frac{derror(t)}{\mathrm{dt}} \tag{4}$$

where Kd is the derivative gain. Increasing Kd makes the control system response more strongly to changes in error signal and raises the speed of overall control system response [10].

The derivative control can improve the system performance as it can anticipate an incorrect trend of the error signal and counteract for it [11,6]. However, the derivative control is extremely sensitive to noise. Therefore, the system can become unstable if there is noise in the sensor feedback signal or the control loop rate is too slow [10].

3 Components

3.1 Sensor units

3.1.1 HC-SR04 Ultrasonic Sensor

The HC-SR04 ultrasonic sensor is a device that is capable of measuring distance to a solid object. Based on the datasheet [12], the theoretical range of its operation is between 2cm and 400cm for non-contact measurement function, with the resolution of 0,3cm. However, the practical optimal performance is between 10cm and 250cm [13].

Figure 23. HC-SR04 ultrasonic sensor [4]

The sensor operation is based on the emission of ultrasonic waves and their return time when bouncing off an object to detect the distance. It includes two main components: a transmitter, which sends a burst of ultrasonic pulses at the frequency of 40kHz, and a

receiver, which listens for the transmitted pulses. As can be seen in figure 23, the HC-SR04 has four following pins:

- VCC – power supply
- Trig – Trigger pin, which is used to send the ultrasonic pulses
- Echo – which produces a pulse when receiving reflected signal
- GND – Ground pin

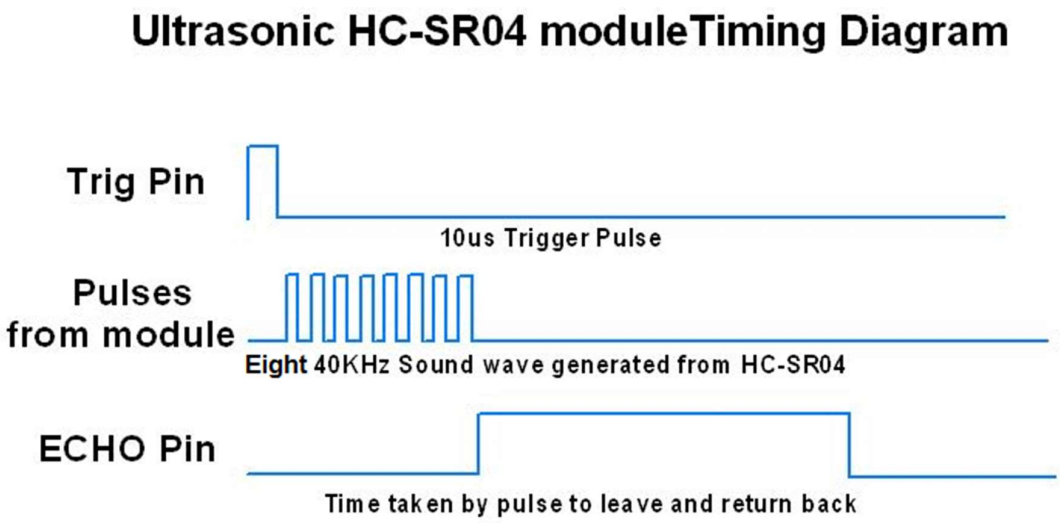

Figure 24. Timing diagram of HC-SR04 ultrasonic sensor [12]

Figure 24 shows the operation of the sensor module. High signal (5V) needs to be applied to the Trigger pin for 10 µs to generate the ultrasonic wave. The module responds by sending a burst of eight pulses at the frequency of 40kHz. Once finished sending the pulses, the echo pin is activated and creates a pulse, which has a width proportional to the distance between the sensor and the detected object. The distance is calculated based on time interval between sending trigger signal and receiving echo signal. Below are two formulas for calculating the distance [12]:

$$distance\ (cm)\ =\ uS(microseconds)/\ 58 \tag{5}$$

or

$$distance\ (cm)\ =\ high\ level\ time * speed\ of\ sound(0.034\ cm/\mu s)/2 \tag{6}$$

Below are some features and electric parameter of the HC-SR04 ultrasonic sensor according to datasheet: [12]

- Working Voltage: DC 5 V
- Working Current: 15mA
- Working Frequency: 40Hz
- Max Range: 4m
- Min Range: 2cm
- Measuring Angle: 15 degree
- Trigger Input Signal: 10uS TTL pulse
- Echo Output Signal: Input TTL level signal and the range in proportion
- Dimension: 45*20*15mm

3.1.2 H206 Opto-Coupler Speed Sensor

The sensor is a photo-interrupter that includes an infrared light-emitting diode (LED) as a source of light and a phototransistor sensor. The infrared LED is placed facing the sensor with a gap of 6 millimeters. In operation, when the gap is clear, the infrared LED shines onto the phototransistor sensor, which senses the light and allows the current passing to emitter from the collector. When a solid non-transparent object is placed in the gap between the infrared LED and the phototransistor sensor, the light beam is blocked, causing the phototransistor to stop passing current. Figure 25 shows two sides of the H206 Opto-Coupler Speed Sensor and the encode the wheel [14.]

Figure 25. H206 Opto-Coupler Speed Sensor sensor and encoder wheel [15]

In this study, the encoder wheel with 20 equal spaced slots is placed in the gap between the infrared LED and the phototransistor sensor. When spinning, the slots of the wheel allows light beam passing through and causes the signal switching on and off according to its rotation. Each pulse represents a slot in the encoder wheel, it means one pulse shows that the wheel has rotated 18 degrees (360 degrees divided by 20).

Additionally, as can be seen in figure 26, an LM393 comparator is mounted into the module to remove error digital signal as the phototransistor creates poor output pulses. The LM393 has two inputs, one for the output of phototransistor sensor and the other for reference voltage. The digital output of LM393 goes high (5V) when the output of phototransistor sensor equals or surpasses the reference voltage. In contrast, it goes low (ground) when the output of phototransistor sensor is below the reference voltage. With LM393 comparator, the H206 Opto-Coupler Speed Sensor module generates more accurate 5V pulse signal, which is used in calculating wheel's speed.

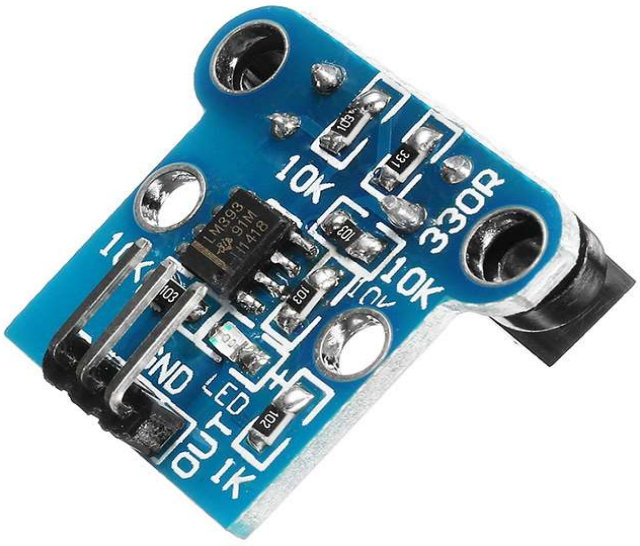

Figure 26. LM393 comparator mounted in the H206 sensor module [15]

By calculating the number of pulses in one minute, it is possible to measure the speed of the wheel in round per minute (rpm) (divided by 20). Besides, measuring the circumference of the wheel helps determine the actual speed in kilometers per hour (km/h) as it is equivalent to the distance travel in one rotation.

The module has three following pins:

- VCC – Power supply
- GND – Ground pin
- OUT – Signal output; standby - output low level, indicator light is on; detected something - output high level, indicator light is off.

Below are some parameters of H206 Opto-Coupler Speed Sensor module [15]:

- Working voltage: DC 4.5-5.5V
- Launch tube current: If <20mA
- Signal output: single channel; TTL level
- Resolution accuracy: 0.01mm
- Long size: 2.6mm x 2.2mm
- The main component: LM393
- Photoelectric slot width: 6mm

3.1.3 Four-channel Infrared Receiver Tracking Sensor

The module includes four sets of infrared transmitter and receiver tracking sensor connected to the hollow board as can be seen in figure 27. Each tracking channel sends high or low signal to the board, depending on the color of surface. The tracking module has twelve input pins for supply voltage, ground and data from four tracking channels. There are also four potentiometers to adjust the sensitivity of four receivers when changing detection range.

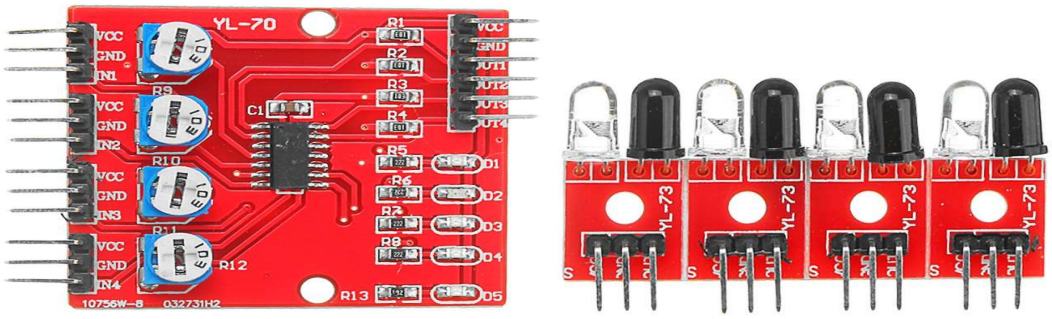

Figure 27. Four-channel infrared receiver tracking sensor [16]

The sensor also has six output pins: VCC for power supply, GND is ground pin and OUT1, OUT2, OUT3, OUT4 for four output signals from four tracking channels. When infrared light emitted from transmitter meets white surface and reflects to the receiver, the indicator LED (D1-4) is on and output signal (OUT1-4) is low. If infrared light is absorbed by black surface, the indicator LED (D1-4) is off and output signal (OUT1-4) is high.

Below are some parameters of the module [16]:

- Working voltage: DC 3.3V-5V
- Working current: try to choose more than 1A power supply
- Working temperature: -10°C to +50°C
- Mounting aperture: M3 screw
- Detection range: 1mm to 60cm adjustable, the performance more stable when closer, white reflects the farthest distance.
- Output interface: 6-wire interface (1234 to 4 signal output ends, + positive power, - for the negative power is ground)
- The output signal: TTL level

3.2 Hardware units

3.2.1 Arduino Uno and Sensor Shield

Arduino Uno is a microcontroller board developed by Arduino.cc, which is an open-source electronics platform built on single-chip microcontroller ATmega328P. It consists of 14 digital input/output pins, 6 of those (D3, D5, D6, D9, D10, D11) can be used to provide 8-bit Pulse Width Modulation (PWM) outputs, 6 analogue pins. The board can be powered by a 5V USB connection or by external power source, which can be up to 12V. [17.]

The board has several features such as timers, counters, interrupts, 16 MHz quartz crystal clock. It also has a reset button, which resets the running program and starts again from the initial stage. Additionally, the board also supports serial communication with Tx (D1) and Rx (D0) pins. [18.]

Arduino IDE (Integrated Development Environment) is a software that is used to transfer the code to the board. This software is compatible with Windows, MAC or Linux Systems and it works with C or C++ language.

The pinout of Arduino Uno board is shown in the figure 28:

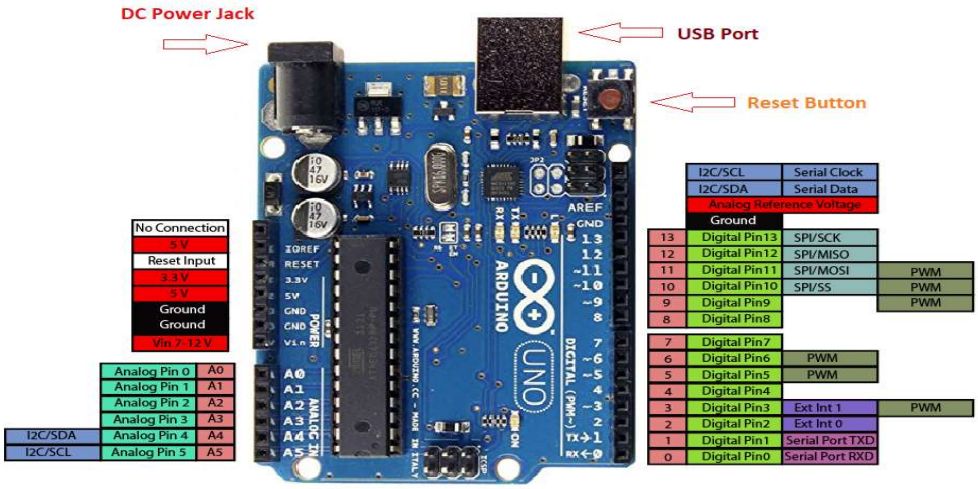

Figure 28. Arduino Uno Pinout [18]

In this study, an Arduino sensor shield is also used to provide the simplified and well-organized wiring connection between sensors and the board.

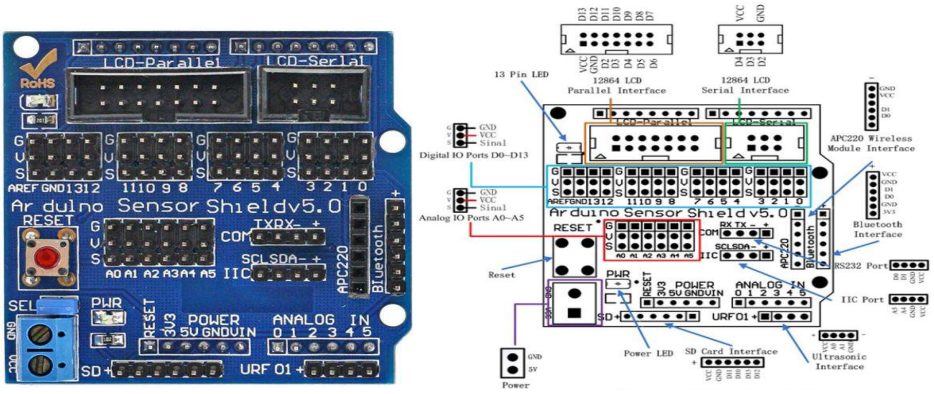

Figure 29. Arduino sensor shield v5.0 and its functional diagram [19]

As can be seen in figure 29, the shield offers one VCC pin (power supply) and one GND pin (ground) for every Arduino signal pin. To provide 5V and ground for every digital and analogue I/O port, after connecting the shield and Arduino Uno board, two SEL pins of the shield need to be connected by a jumper. If those two SEL pins are not connected, only analogue I/O ports get 5V and ground from Arduino Uno board. In this case, digital I/O ports get VCC and GND from external power source that connects directly to the shield.

Below are some parameters of Arduino Uno module [17]:

- Microcontroller: ATmega328P – 8-bit AVR family microcontroller
- Working voltage: 5V
- Recommended Input Voltage: 7-12V
- Input Voltage Limits: 6-20V
- Analog Input Pins: 6 (A0 – A5)
- Digital I/O Pins: 14 (Out of which 6 provide PWM output)
- DC Current on I/O Pins: 40 mA
- DC Current on 3.3V Pin: 50 mA
- Flash Memory: 32 KB
- Frequency (Clock Speed): 16 MHz

3.2.2 L298N Motor Driver

The L298N is a dual H-Bridge motor driver which can control the direction and speed of two DC motors at the same time. DC motors (5V to 35V) can be driven by the module with peak current up to 2A. [20.]

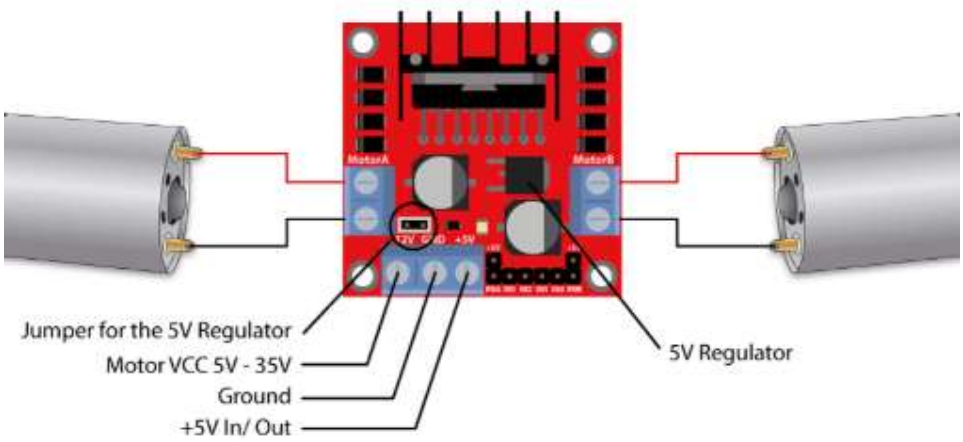

Figure 30. L298N motor driver [20]

As can be seen in figure 30, the motor has two screw terminal blocks for driving two different motors, another 3-screw terminal block for supply voltage, ground pin and a 5V pin which can be used as either input or output. The module also has a jumper to active a 5V regulator onboard. If the supply voltage is below 12V, the regulator can be activated to provide 5V pin as an output. On the other hand, if the supply voltage is greater than 12V, the jumper needs to be removed in order not to damage the regulator. In this case, the 5V pin becomes an input and requires a 5V power supply to help the module operate properly.

Besides, there are six logic control inputs. The ENA and ENB pins are used to activate and control motors' speed. If the pin is connected by the jumper, the motor will be active and operate at full speed. If the jumper is removed, it is possible to control the motor's speed by connecting the pin to a PWM pin of Arduino. Connecting this pin to ground will disable the motor. The IN1 and IN2 pins are used for controlling the rotation direction of Motor A, and IN3 and IN4 are for Motor B.

Below are some parameters of the module [21]:

- Double H bridge drive
- Chip: L298N
- Logical voltage: 5V

- Drive voltage: 5V-35V
- Logical current: 0mA-36mA
- Drive current: 2A (MAX single bridge)
- Storage temperature: -20°C to +135°C
- Max power: 25W
- Weight: 30g
- Size: 43 x 43 x 27mm

4 Design of the prototypes

The goal of this study was to create a safety system, which can stop a moving robot when an object is detected within a safety distance that is dependent on the velocity of the robot. Therefore, it is important to first determine precisely the robot's distance to the obstacle around and its velocity. Information about distance and velocity is shown on a liquid-crystal display (LCD). Based on that information, each prototype is designed and programmed to operate as its function. First prototype is a directional robot controlled by a joystick and a second one is a black line robot follower. Once an object is detected within a safety distance, they will stop moving forward until their safety range is clear. In this study, two cases of safety distance are defined by the velocity of the robot. When it is in faster mode, the safety distance is set to 20 centimeters. When it is in slower mode, the safety distance is set to 10 centimeters.

4.1 Joystick-Control Prototype

This prototype is a robot with a joystick to control its speed and direction. A joystick is created by two potentiometers, which represent horizontal and vertical axes, and are connected to the analogue pins of Arduino to control the speed of two motors. Two motors have the PWM speed value that varies from 0 (stop) to 255 (maximum speed) and can change rotation direction depending on the current flow, which is controlled by the joystick.

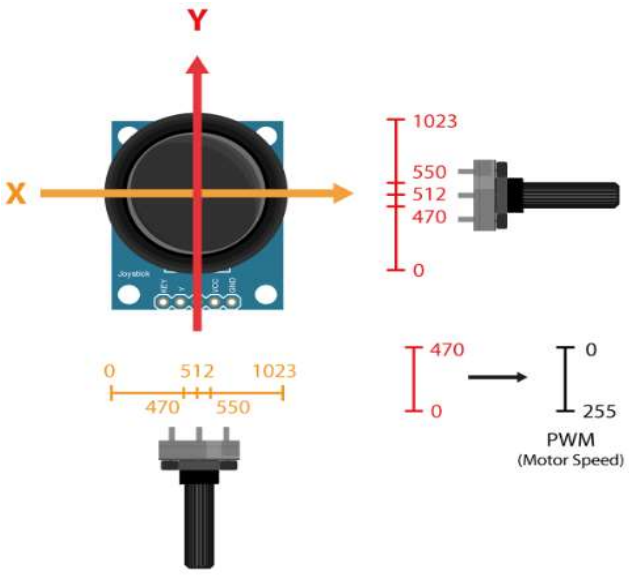

Figure 31. Joystick diagram [20]

As can be seen in figure 31, the value of both potentiometers ranges from 0 to 1023. When the joystick is at default position, both axes have a value of 512 roughly and the speed of two motors is zero. With Y axis, when the position of the joystick is from 550 to 1023, two motors move forward with the speed varies corresponding from 0 to 255. Similarly, when the position of the joystick is from 470 to 0, two motors move backward with the speed varies corresponding from 0 to 255.

Moving the joystick in X axis controls the turning of the robot. When the position of the joystick in X axis varies from 470 to 0, it increases the speed of the right motor corresponding from 0 to 255, which makes the robot turn left. In contrast, when the position of the joystick in X axis varies from 550 to 1023, it increases the speed of the left motor corresponding from 0 to 255, which makes the robot turn right.

Figure 32 illustrates the front view and top view of the Joystick-control prototype.

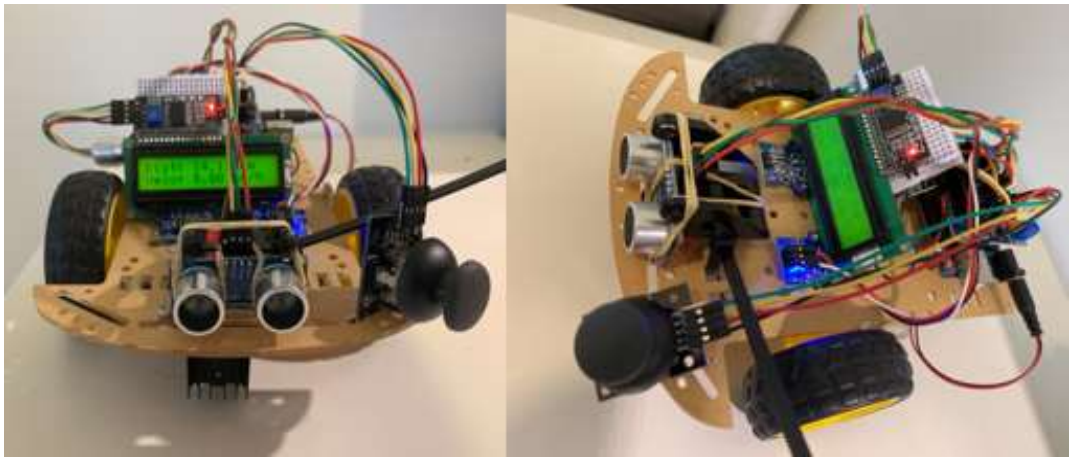

Figure 32. Joystick-control prototype

As can be seen in figure 34, the Arduino sensor shield is connected to and directly powered by Arduino Uno board. In this study, a 9V battery is used to supply power to Arduino Uno board, and it offers 5V to every V output pin of the shield. Firstly, HC-SR04 ultrasonic sensor is connected to the shield for distance detection. Trig and Echo pin are connected to pin D5, D4 in the shield respectively. Based on the application note of the sensor, the Ground pin should be connected before VCC pin. The Trig pin is set to LOW state for 2 µs and then it is set to HIGH state for 10 µs to send the ultrasonic waves. After that, calculating the HIGH state time of Echo pin gives the duration that ultrasonic waves travel to an object and reflect, then dividing by 2 and multiplying with 0.0343 (343 meters per second as speed of sound) will give the distance from the sensor to the object in centimeter.

Secondly, two motors are connected to L298N motor driver, and two H206 Opto-Coupler Speed Sensor are connected to the Arduino shield to measure the speed of the wheels, which are attached to the motors. VCC and GROUND pin are hooked up to the shield while OUTPUT pin of the left sensor is connected to pin D3 of Arduino Uno, which is an interrupt pin. There are two encoder disks, which have 20 equal spaced slots, attached to two motors and placed in the gap position of the sensors as can be seen in figure 33.

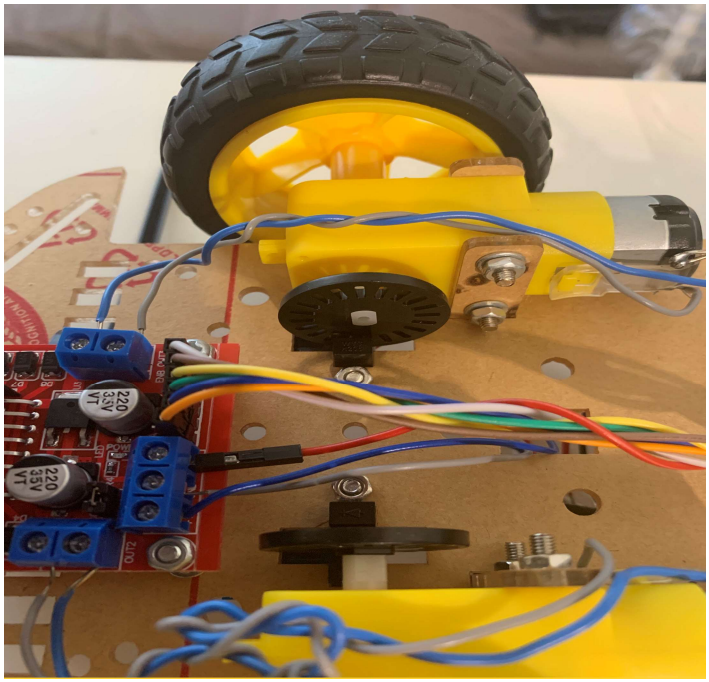

Figure 33. Speed sensor in place

When the motor rotates, the disk will rotate with the same speed and its holes create interrupts, which happen when the state of the sensor digital output changes. An Arduino function called Interrupt Service Routine (ISR) is used to detect when an interrupt happens. Dividing the number of interrupts in one minute by 20 will give the speed of the wheel in round per minute. Tape is wrapped around the outside of the wheel and its measurement is 21 centimeters. This is used to provide the speed of the wheel in centimeters per minute and convert to kilometer per hour by dividing by 1666.67.

Thirdly, a joystick is connected to the shield as can be seen in figure 34. VRx and VRy are connected respectively to A0 and A1 pin of the shield to provide analogue signals to drive the motors.

Then another 9V battery is used to power the L298N motor driver and the 5V pin also needs to be connected to any VCC pin of the shield. ENA and ENB pin of the driver are hooked up respectively to pin D9 and D11, which are PWM pins to control the speed of two motors. IN1, IN2, IN3, IN4 pin of the driver, which are used to control the rotation direction of two motors, are connected correspondingly to pin D2, D8, D13, D12.

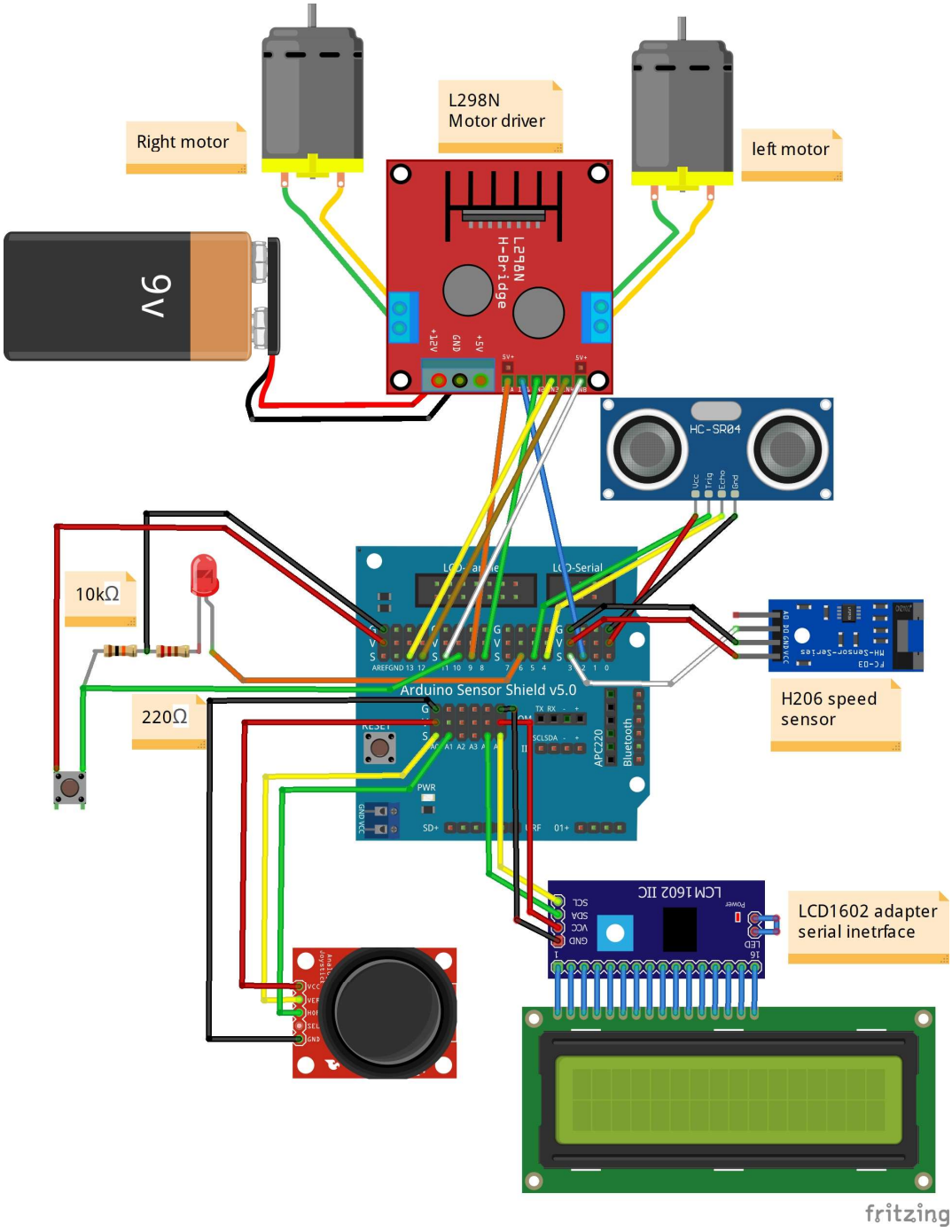

Figure 34. Joystick-control prototype circuit wiring made with fritzing program

An LCD1602 is connected to the shield via a serial interface adapter to display the velocity of the robot and the distance between the robot to the object in front of it. The advantage of the adapter is it only takes two analogue pins as SDA pin and SCL pin are connected to pin A4 and A5, while a normal LCD can take up to six digital pins of the Arduino to operate.

Finally, to set up two different safety distances for the prototype depending on its velocity, a Boolean variable is created in the program. It has a default value of 0 for the velocity that is below 5 km/h, while the safety distance is set at 10 centimeters. When the velocity is greater than 5 km/h, the variable changes its value to 1 and sets the safety distance to 20 centimeters, it also lights up the LED. A button is also connected as showed in figure 34 to switch the speed mode. Once the data of velocity and distance is acquired, it is possible to determine when to stop the robot from moving forward and the source code is provided in the Appendix 1.

4.2 Line Follower Prototype

The prototype is a black line robot follower in white surface. Like the joystick-control prototype, it also includes a HC-SR04 ultrasonic sensor, two H206 Opto-Coupler Speed Sensors for distance and speed calculation, and a LCD1602 with its serial interface adapter to display those values. Figure 35 illustrates the front view and top view of the line follower prototype.

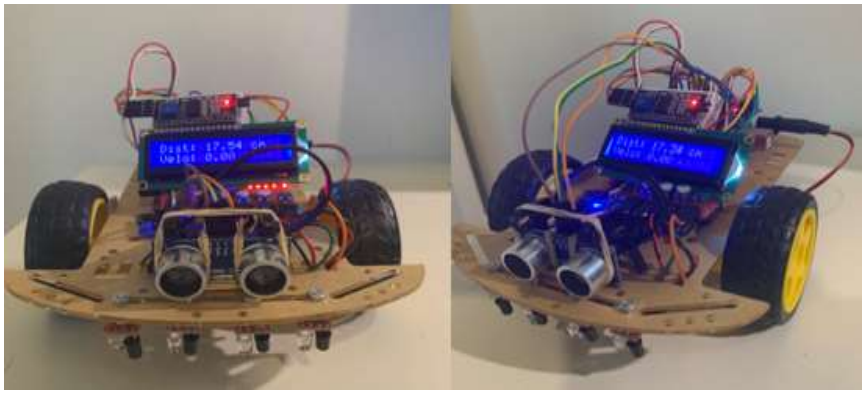

Figure 35. Line follower prototype

The wiring connection can be seen in figure 36. Trig pin, Echo pin of HC-SR04, and output pin of H206 are connected respectively to pin D7, D6, D2, while SDA and SCL pin of the adapter are hooked to pin A4, A5 of the Arduino shield.

Two motors are also connected to L298N driver, which is powered by a 9V battery. In this prototype, ENA and ENB pin of the L298N also need to be connected to PWM pin, and EN1, EN2, EN3, EN4 pin are hooked to analogue pins of the Arduino shield. The L298N will drive two motors based on the digital signals from four IR tracking sensors.

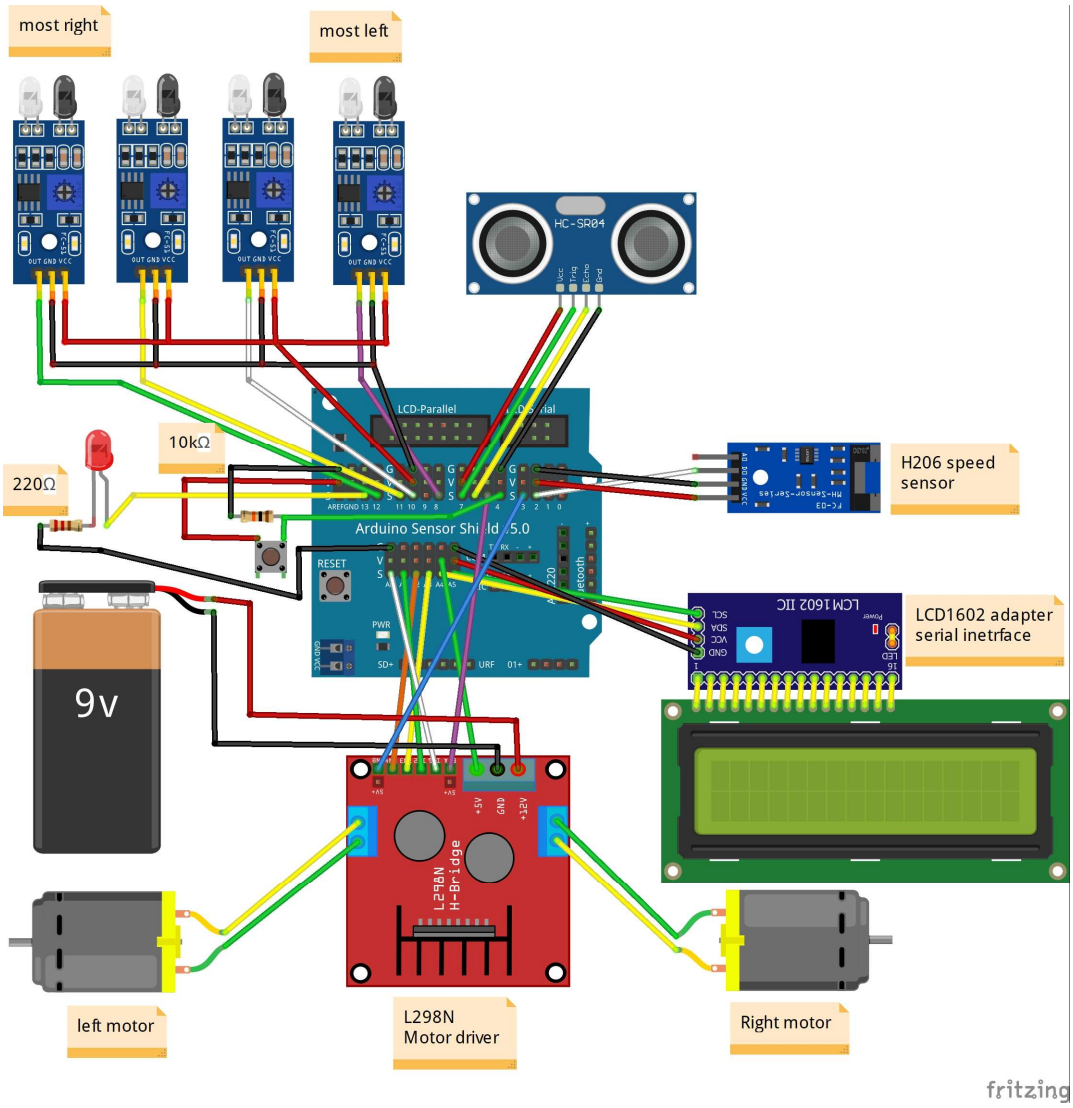

Figure 36. Line follower prototype circuit wiring made with fritzing program

It is important to adjust the potentiometers so that each tracking sensor can return accurate signal corresponding to the color of the surface. When facing a black surface, the sensor turns off its LED and provide high level digital output. In contrast, when facing a white surface, the sensor turns on its LED and provide low level digital output. From the most left to the most right, the sensors are connected respectively to pin D8, D10, D11 and D12. Figure 37 describes eleven cases that the prototype may face.

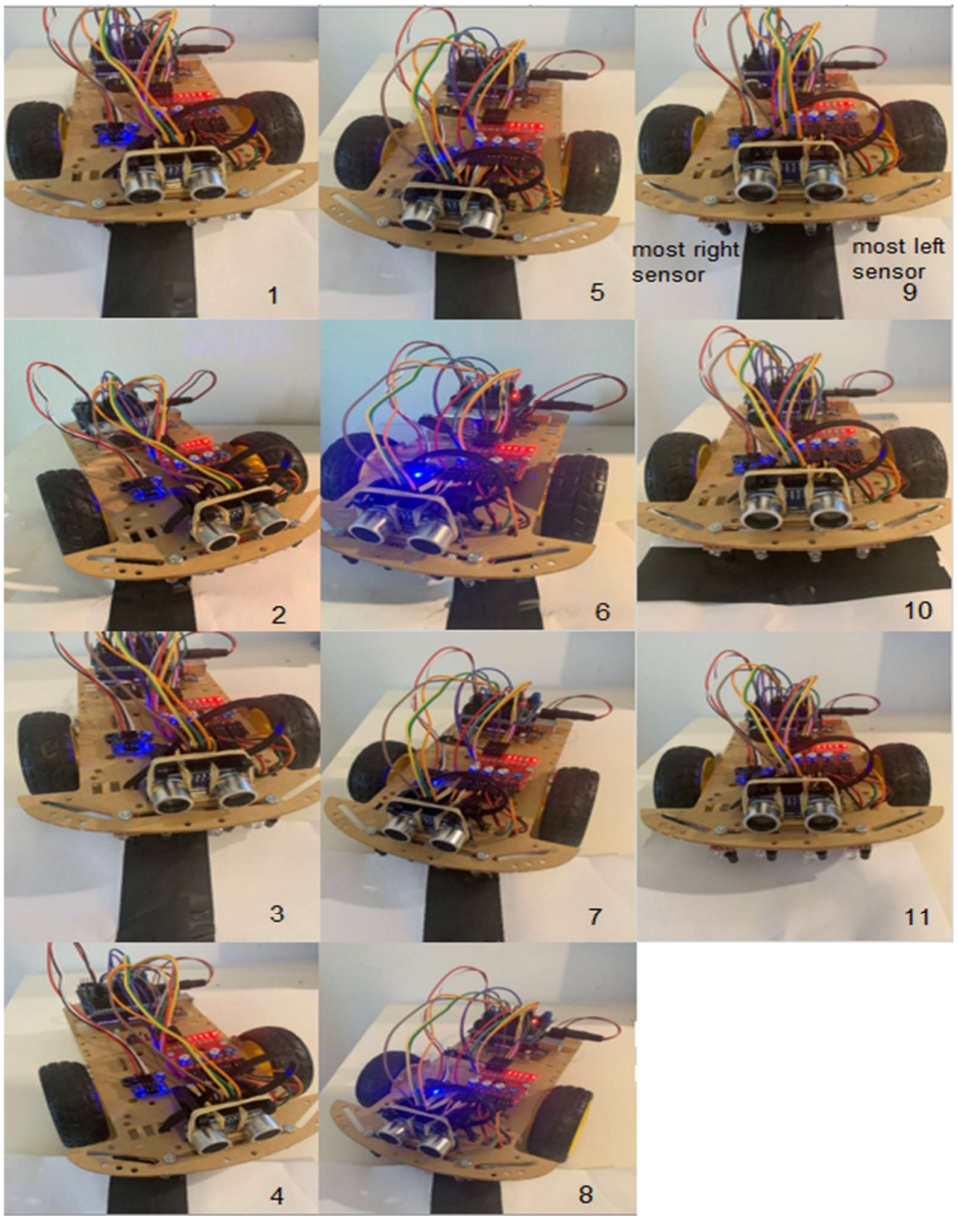

Figure 37. Eleven cases of the line follower prototype

As can be seen in figure 37, depending on the position of the robot to the black line, four tracking sensors create different combination of four digital outputs. The table below shows more detail of these combinations.

	Most right sensor			Most left sensor		
Case	**D12**	**D11**	**D10**	**D8**	**Robot next action**	**Error**
1	0	1	0	0	turn right	1
2	1	0	0	0	turn right	2
3	1	1	0	0	turn right	3
4	1	0	0	0	turn sharp right	4
5	0	0	1	0	turn left	-1
6	0	0	0	1	turn left	-2
7	0	0	1	1	turn left	-3
8	0	0	0	1	turn sharp left	-4
9	0	1	1	0	keep straight	0
10	1	1	1	1		0
11	0	0	0	0		0

Table 2. Eleven cases of the line follower prototype

The table 2 illustrates eleven cases of four digital signals combination that Arduino receives based on the position of the robot to the black line and how the robot reacts. Additionally, it also shows the $error$ values, which are used for PID control as:

$$P = Kp * error \tag{7}$$

$$I = Ki * integral \tag{8}$$

$$D = Kd * derivative \tag{9}$$

$$output = P + I + D \tag{10}$$

where $integral$ is the sum of all previous $error$ values, $derivative$ is the difference between current $error$ value and previous $error$ value. The $output$ value is used to adjust the speed of two motors to minimize the $error$ value to zero, which means to keep the

robot moving on the black line. After testing the robot, Kp, Ki and Kd values are tuned to 15, 0 and 20 to maintain the stability of robot movement.

Finally, distance and velocity of the robot are calculated by the same methods that are used in the joystick-control prototype. There are two speed modes for this prototype, which can be switched by a button. In the first mode, the speed of motors is set to fast speed level, safety distance is set at 20 centimeters and the LED is on. In the second mode, the speed of motors is set slower, safety distance is changed to 10 centimeters and the LED is off. A button is also used to switch between two modes. Once the data of velocity and distance is acquired, it is possible to determine when to stop the robot from moving forward and the source code is provided in the Appendix 2.

5 Conclusion

To sum up, the purpose of this project was to design a safety system for MHV prototypes, which can stop the robots from moving forward if an object is detected within a safety distance. The safety distance changes depending on the speed of the robots, it is further with faster speed. By applying theoretical knowledge of ultrasonic waves and photoelectric sensor, the prototype can detect the distance from itself to an object using HC-SR04 sensor and determine its velocity with H206 sensor. Collected data are processed with Arduino Uno to control the motors via L298N driver.

In general, two prototypes successfully work as desired. Both can detect and display their distance to any object as well as their velocity. When any object is placed within their safety range they will stop moving forward until the object is removed. With the Joystick-control prototype, it is possible to control the moving direction and the speed of the robot. The safety system can set up two different safety distances for the velocity below and greater than 5 km/h. The line follower prototype can move along the black line on a white surface. It has two speed modes and the safety system can separate two different safety distances. The robot might not be able to follow complicated routes because this project is mainly focused on the safety system.

However, the prototypes are two simple robots made with affordable components and sensors. Therefore, plenty of improvements can be made for further development. For example, extra sensors can be added to the sides as well as the back of the robot to cover larger safety range. Besides, due to its limitations, HC-SR04 sensor can be replaced by Light Detection and Ranging (LiDAR) sensor to increase the accuracy in distance reading. Moreover, better techniques of speed-sensing should be implemented to provide more precise velocity measurement. With more sensors, it is also necessary to replace Arduino Uno by another microcontroller with a greater number of pinouts. Especially in this study, another power source such as Lithium Polymer (Li-Po) battery can be a better option as a 9V PP3 battery does not offer long battery life. Above are few possible developments that can be made to improve the performance of the system. Further improvements may help the system become more practical and can be applied in real industrial MHV.

References

1. **Statistics Finland.** Self-employed persons' accidents at work. *Statistics Finland.* [Online] November 30, 2012. [Cited: May 22, 2019.] https://tilastokeskus.fi/til/ttap/2010/ttap_2010_2012-11-30_kat_002_en.html

2. **Fraden, Jacob.** *Handbook of modern sensors: physics, designs, and applications Fourth Edition.* New York: Springer Science+Business Media, 2010. ISBN 978-1-4419-6465-6

3. **Seebo Blog.** Ultrasonic Sensors: Applications in the Internet of Things. *Seebo Blog web site.* [Online] Seebo Blog, May 28, 2018. [Cited: May 28, 2019.] https://blog.seebo.com/iot-ultrasonic-sensors/

4. **Lucian, Vaduva Ionut.** ARDUINO SNOW DEPTH REMOTE SENSING WITH ULTRASONIC SENSOR. *GeeksTips.* [Online] [Cited: 29 July 2019.] https://www.geekstips.com/arduino-snow-depth-remote-sensing-with-ultrasonic-sensor/

5. **Transparency Market Research.** Photoelectric Sensors Market. *Transparency Market Research.* [Online] May 2016. [Cited: June 4, 2019.] https://www.transparencymarketresearch.com/photoelectric-sensors-market.html

6. **OMRON Industrial Automation.** Photoelectric Sensors. *OMRON Industrial Automation.* [Online] OMRON Industrial Automation. [Cited: June 5, 2019.] http://www.ia.omron.com/support/guide/43/introduction.html

7. **ifm electronic.** Training manual photoelectric sensors. *ifm electronic.* [Online] March 2003. [Cited: June 5, 2019.] www.ifm.com/obj/s200e.pdf

8. **SoftNoze USA Inc.** Photoelectric Sensors Theory of Operation. *SoftNoze.* [Online] [Cited: June 7, 2019.] www.softnoze.com/downloads/Sensor%20Basics%204.pdf

9. **Saddam.** Line Follower Robot using Arduino. *Circuit Digest.* [Online] 18 June 2015. [Cited: 4 July 2019.] https://circuitdigest.com/microcontroller-projects/line-follower-robot-using-arduino

10. **National Instruments.** PID Theory Explained. *National Instruments.* [Online] National Instruments, March 5, 2019. [Cited: July 30, 2019.] http://www.ni.com/fi-fi/innovations/white-papers/06/pid-theory-explained.html

11. **Visioli, Antonio.** *Practical PID Control.* Brescia : Springer, 2006. ISBN-13: 9781846285851

12. **Elecfreaks.** Ultrasonic Ranging Module HC - SR04. *Elecfreaks.* [Online] [Cited: 22 July 2019.] https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

13. **Bermudez, Brayan.** sensor ultrasonido hc-sr04 con psoc5lp. *PSoCLatinoamerica.* [Online] 24 November 2016. [Cited: 22 July 2019.] https://blog.psoclatinoamerica.co/sensor-ultrasonido-hc-sr04-con-psoc5lp/

14. **DroneBot Workshop.** Build a Robot Car with Speed Sensors. *DroneBot Workshop.* [Online] DroneBot Workshop, December 8, 2017. [Cited: July 23, 2019.] https://dronebotworkshop.com/robot-car-with-speed-sensors/

15. **Banggood.** H206 Photoelectric Counter Counting Sensor Module Motor Speed Board Robot Speed Code 6MM Slot Width. *Banggood.* [Online] Banggood. [Cited: July 30, 2019.] https://www.banggood.com/H206-Photoelectric-Counter-Counting-Sensor-Module-Motor-Speed-Board-p-1204425.html?utm_design=41&utm_source=emarsys&utm_medium=Shipoutinform171129&utm_campaign=trigger-emarsys&utm_content=Winna&sc_src=email_2671705&sc_eh=d0abf4d71

16. **Banggood.** 4 Channel Infrared Receiver Tracking Sensor Module IR Line Patrol For Arduino. *Bang good.* [Online] Bang good. [Cited: 23 July 2019.] https://www.banggood.com/4-Channel-Infrared-Receiver-Tracking-Sensor-Module-IR-Line-Patrol-For-Arduino-p-1414298.html?utm_design=41&utm_source=emarsys&utm_medium=Shipoutinform171

129&utm_campaign=trigger-emarsys&utm_content=Winna&sc_src=email_2671705&sc_eh

17. **Arduino.cc.** ARDUINO UNO REV3. *Arduino.cc.* [Online] Arduino.cc. [Cited: July 30, 2019.] https://store.arduino.cc/arduino-uno-rev3

18. **Aqeel, Adnan.** Introduction to Arduino Uno. *The Engineering Projects.* [Online] The Engineering Projects, 21 June 2018. [Cited: 24 July 2019.] https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-uno.html

19. **el gammal electronics.** Arduino sensor Shield V5.0 . *el gammal electronics.* [Online] [Cited: July 30, 2019.] http://www.elgammalelectronics.com/Products/Download/f18faaa0-8a40-493b-b754-8b33a7a06b59?Name=XX30-Arduino%20Sensor%20Shield%20V5.0%20sensor%20expansion%20board%20electronic%20building%20blocks%20new-DATASHEET

20. **Dejan.** Arduino DC Motor Control Tutorial – L298N | PWM | H-Bridge. *HowToMechatronics.* [Online] HowToMechatronics. [Cited: 25 July 2019.] https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/

21. **Geekcreit.** Banggood. *Banggood.* [Online] Banggood. [Cited: July 30, 2019.] https://www.banggood.com/Wholesale-Dual-H-Bridge-DC-Stepper-Motor-Drive-Controller-Board-Module-Arduino-L298N-p-42826.html?utm_design=41&utm_source=emarsys&utm_medium=Shipoutinform171129&utm_campaign=trigger-emarsys&utm_content=Winna&sc_src=email_2671705&

Appendix 1. Joystick-control prototype code

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <NewPing.h>
#include <TimerOne.h>

#define TRIGGER_PIN  5
#define ECHO_PIN     4
#define MAX_DISTANCE 400
#define buttonPin 10
#define ledPin 6

LiquidCrystal_I2C lcd(0x27, 16, 2);

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

float duration;
float distance;                 // in cm
float diskslots = 20;           // encoder plate has 20 holes
float rotation;
float velocity;
float p = 21;                   // perimeter of the wheel in cm
int   flag = 0;
int   buttonState = 0;
const byte   MOTOR = 3;  // motor interrupt pin
unsigned int counter = 0;  // integers for pulse counters

// Motor A - Right
int enA = 9;
int in1 = 2;
int in2 = 8;
```

```
// Motor B - Left
int in3 = 13;
int in4 = 12;
int enB = 11;

int joyY = A1;  // Vertical Y
int joyX = A0;  // Horizontal X

int MotorSpeed1 = 0;
int MotorSpeed2 = 0;

// Joystick Values - Start at 512 (middle position)
int joyposY = 512;
int joyposX = 512;

void ISR_count()
{
  counter++;
}

void ISR_timerone()
{
  Timer1.detachInterrupt();                  // stop the timer
  rotation = (counter / diskslots) * 60.00;  // calculate RPM for Motor
  velocity = rotation * p / 1666.67;         // convert RPM to cm/m and then to km/h
  counter = 0;                               // reset counter to zero
  Timer1.attachInterrupt( ISR_timerone );  // enable the timer
}

void setup()
{
  Serial.begin(9600);
  lcd.begin();
  lcd.backlight();
```

```
  Timer1.initialize(1000000); // set timer for 1sec
  attachInterrupt(digitalPinToInterrupt (MOTOR), ISR_count, RISING);  // Increase counter when speed sensor pin goes High
  Timer1.attachInterrupt( ISR_timerone ); // Enable the timer

  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);

  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);

  digitalWrite(ledPin, LOW);

  // Motor A - Right
  digitalWrite(enA, LOW);
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);

  // Motor B - Right
  digitalWrite(enB, LOW);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
}

void loop()
{
  joyposY = analogRead(joyY);
  joyposX = analogRead(joyX);

  buttonState = digitalRead(buttonPin);
```

```
 if (velocity > 5)
  {
  digitalWrite(ledPin,HIGH);
  flag = 1;
  }

 if (buttonState == HIGH)
  {
  digitalWrite(ledPin, LOW);
  flag = 0;
  }

 if  (((joyposY < 460) && (flag == 0) && (distance > 10)) || ((joyposY < 460) && (flag ==
1) && (distance > 20)) )
 {
  // Move forward

  // Set right Motor forward
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);

  // Set left Motor forward
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);

  //Determine Motor Speeds
  joyposY = joyposY - 460;   // This produces a negative number
  joyposY = joyposY * -1;     // Make the number positive

  MotorSpeed1 = map(joyposY, 0, 460, 0, 255);
  MotorSpeed2 = map(joyposY, 0, 460, 0, 255);

  rotation = rotation;
 }
```

```
else if (joyposY > 564)
{
  // Move backward

  // Set right Motor backward
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);

  // Set left Motor backward
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);

  //Determine Motor Speeds
  MotorSpeed1 = map(joyposY, 564, 1023, 0, 255);
  MotorSpeed2 = map(joyposY, 564, 1023, 0, 255);
}

else
{
  MotorSpeed1 = 0;
  MotorSpeed2 = 0;
}

if (joyposX < 460)
{
  // Move left

  joyposX = joyposX - 460;
  joyposX = joyposX * -1;
  joyposX = map(joyposX, 0, 460, 0, 255);
```

```
  MotorSpeed2 = MotorSpeed2 - joyposX;
  MotorSpeed1 = MotorSpeed1 + joyposX;

  if (MotorSpeed2 < 0) MotorSpeed2 = 0;
  if (MotorSpeed1 > 255) MotorSpeed1 = 255;
}

else if (joyposX > 564)
{
  // Move right
  joyposX = map(joyposX, 564, 1023, 0, 255);

  MotorSpeed2 = MotorSpeed2 + joyposX;
  MotorSpeed1 = MotorSpeed1 - joyposX;

  if (MotorSpeed2 > 255) MotorSpeed2 = 255;
  if (MotorSpeed1 < 0) MotorSpeed1 = 0;
}

// Prevent motor buzzing at very low speed
if (MotorSpeed1 < 8) MotorSpeed1 = 0;
if (MotorSpeed2 < 8) MotorSpeed2 = 0;

// Set the motor speeds
analogWrite(enA, MotorSpeed1);
analogWrite(enB, MotorSpeed2);

duration = sonar.ping();                // Determine distance from duration
distance = (duration / 2) * 0.0343;  // Use 343 meters per second as speed of sound

lcd.clear();
lcd.print("Dist: ");
lcd.setCursor (6,0);
lcd.print(distance);
```

```
lcd.print(" cm");
lcd.setCursor (0,1);
lcd.print("Velo: ");
lcd.setCursor (6,1);
lcd.print(velocity);
lcd.print(" km/h ");

delay(500);
```

Appendix 2. Line Following Prototype code

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <NewPing.h>
#include <TimerOne.h>

#define TRIGGER_PIN  7
#define ECHO_PIN     6
#define MAX_DISTANCE 400
#define buttonPin 4
#define ledPin 13

LiquidCrystal_I2C lcd(0x27, 16, 2);

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

float duration;
float distance;              // in cm
float diskslots = 20;        // encoder plate has 20 holes
float rotation;
float velocity;
int flag = 1;
int buttonState = 0;

float safe = 20;

float p = 21;                 // perimeter of the wheel in cm
const byte   MOTOR = 2;   // motor interrupt pin
unsigned int counter = 0;   // integers for pulse counters

float pTerm, iTerm, dTerm;
int error;
```

```
int previousError;
float kp = 15;
float ki = 0;
float kd = 20;

float output;
int integral, derivative;
int irSensors[] = {12, 11, 10, 8}; //IR sensor pins
int irReadings[4];

int motor1Forward = A1;
int motor1Backward = A0;
int motor1pwmPin = 5;
int motor2Forward = A2;
int motor2Backward = A3;
int motor2pwmPin = 3;

int motor1newSpeed;
int motor2newSpeed;
int motor2Speed = 70;
int motor1Speed = 125;

void ISR_count()
{
  counter++;
}
void ISR_timerone()
{
  Timer1.detachInterrupt();                  // stop the timer
  rotation = (counter / diskslots) * 20.00;  // calculate RPM for Motor
  velocity = rotation * p / 1666.67;         // convert RPM to cm/m and then to km/h
  counter = 0;                               // reset counter to zero
  Timer1.attachInterrupt( ISR_timerone ); // enable the timer
}
```

```
void setup() {
  Serial.begin(9600);
  lcd.begin();
  lcd.backlight();

  Timer1.initialize(3000000); // set timer for 3sec
  attachInterrupt(digitalPinToInterrupt (MOTOR), ISR_count, RISING);  // Increase coun-
ter when speed sensor pin goes High
  Timer1.attachInterrupt( ISR_timerone ); // Enable the timer

  for (int pin = 0; pin < 4; pin++) {
    int pinNum = irSensors[pin];
    pinMode(pinNum, INPUT);
  }

  pinMode(motor1Forward, OUTPUT);
  pinMode(motor1Backward, OUTPUT);
  pinMode(motor1pwmPin, OUTPUT);
  pinMode(motor2Forward, OUTPUT);
  pinMode(motor2Backward, OUTPUT);
  pinMode(motor2pwmPin, OUTPUT);

  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);

  digitalWrite(ledPin, HIGH);
}
 void readIRSensors() {
  for (int pin = 0; pin < 4; pin++) {
    int pinNum = irSensors[pin];
    irReadings[pin] = digitalRead(pinNum);
  }
}
```

```
void calculateError() {
  if ((irReadings[0] == 0) && (irReadings[1] == 1) && (irReadings[2] == 1) &&
(irReadings[3] == 1)) {
    error = -4;
  } else if ((irReadings[0] == 0) && (irReadings[1] == 0) && (irReadings[2] == 1) &&
(irReadings[3] == 1) ) {
    error = -3;
  } else if ((irReadings[0] == 0) && (irReadings[1] == 0) && (irReadings[2] == 0) &&
(irReadings[3] == 1) ) {
    error = -2;
  } else if ((irReadings[0] == 0) && (irReadings[1] == 0) && (irReadings[2] == 1) &&
(irReadings[3] == 0) ) {
    error = -1;
  } else if ((irReadings[0] == 0) && (irReadings[1] == 1) && (irReadings[2] == 1) &&
(irReadings[3] == 0) ) {
    error = 0;
  } else if ((irReadings[0] == 0) && (irReadings[1] == 1) && (irReadings[2] == 0) &&
(irReadings[3] == 0) ) {
    error = 1;
  } else if ((irReadings[0] == 1) && (irReadings[1] == 0) && (irReadings[2] == 0) &&
(irReadings[3] == 0) ) {
    error = 2;
  } else if ((irReadings[0] == 1) && (irReadings[1] == 1) && (irReadings[2] == 0) &&
(irReadings[3] == 0) ) {
    error = 3;
  } else if ((irReadings[0] == 1) && (irReadings[1] == 1) && (irReadings[2] == 1) &&
(irReadings[3] == 0) ) {
    error = 4;
  } else if ((irReadings[0] == 0) && (irReadings[1] == 0) && (irReadings[2] == 0) &&
(irReadings[3] == 0) ) {
    error = 0;
  } else if ((irReadings[0] == 1) && (irReadings[1] == 1) && (irReadings[2] == 1) &&
(irReadings[3] == 1) ) {
    error = 0;
```

```
  }
}

void pidCalculations() {
  pTerm = kp * error;
  integral += error;
  iTerm = ki * integral;
  derivative = error - previousError;
  dTerm = kd * derivative;
  output = pTerm + iTerm + dTerm;
  previousError = error;
}

void changeMotorSpeed() {
  motor2newSpeed = motor2Speed + output;
  motor1newSpeed = motor1Speed - output;

  constrain(motor2newSpeed, 0, 255);
  constrain(motor1newSpeed, 0, 255);

  analogWrite(motor2pwmPin, motor2newSpeed);
  analogWrite(motor1pwmPin, motor1newSpeed);
  digitalWrite(motor2Forward, HIGH);
  digitalWrite(motor2Backward, LOW);
  digitalWrite(motor1Forward, HIGH);
  digitalWrite(motor1Backward, LOW);

  if (distance < safe) {
    analogWrite(motor2pwmPin, 0);
    analogWrite(motor1pwmPin, 0);
    digitalWrite(motor2Forward, LOW);
    digitalWrite(motor2Backward, LOW);
    digitalWrite(motor1Forward, LOW);
    digitalWrite(motor1Backward, LOW);
```

```
  }
}

void loop() {
  buttonState = digitalRead(buttonPin);

  if (flag == 1){
    digitalWrite(ledPin,HIGH);
    safe = 20;
    motor2Speed = 70; //Default faster speed
    motor1Speed = 125; //Default faster speed
  }

  if (buttonState == HIGH) {
    digitalWrite(ledPin, LOW);
    safe = 10;
    flag = 0;
    motor2Speed = 60; //Set lower speed
    motor1Speed = 90; //Set lower speed
  }

  duration = sonar.ping(); // Determine distance from duration
  distance = (duration / 2) * 0.0343; // Use 343 metres per second as speed of sound

  readIRSensors();
  calculateError();
  pidCalculations();
  changeMotorSpeed();

  lcd.clear();
  lcd.print("Dist: ");
  lcd.setCursor (6,0);
  lcd.print(distance);
  lcd.print(" cm");
```

```
  lcd.setCursor (0,1);
  lcd.print("Velo: ");
  lcd.setCursor (6,1);
  lcd.print(velocity);
  lcd.print(" km/h ");
}
```